*Développement de la partie FRONT de l’application NUTRAPP*

Le framework utilisé pour le front est **Angular 6** et **Angular Material**.

Le lien du projet github est le suivant : <https://github.com/Super-meat-boy/Nutrapp>

**Sur Angular il a été déployé 8 components:**

*1/****L’authentification:***

Aucun développement n’a été effectué sur cette partie.

*2/****Contact-us:***

Il correspond à la page “Contact-us” de l’application Nutrapp.

Sur la partie HTML: Une zone de saisie de l’email + une zone de commentaire a été ajoutée à la page.

Sur la partie TS: Il a été ajouté une vérification de l’email avec un message en cas d’erreur de saisie.

*3/****Create-profile:***

Il correspond à la page “My profil” de l’application Nutrapp.

Sur la partie HTML: il a été déployé des zones de texte permettant à l’utilisateur de saisir son nom, prénom, âge,poids, mail, date anniversaire et un bouton “submit”.

*4/****favorite-meals:***

Il correspond à la page “My favorite-meals.

Sur la partie TS: mise en place d’une méthode displayMenus permettant de récupérer en base de données les menus saisis et enregistrés par l’utilisateur dans “Glycemic charge calculator”.

Dans la partie HTML: affichage en enfilade des aliments présents dans chaque menu et de la charge glycémique.

5/***footer:***

Il correspond au pied de page pour l’ensemble des pages de l’application, avec ajout de boutons permettant une redirection vers facebook, twitter, et instagram.

6/***glycemic-load:***

Il correspond à la page “Glycemic charge calculator”.

*Pour la partie TS:*

⇒ déploiement d’une méthode “calculTotalCharge” permettant de calculer la charge totale glycémique d’un repas.

⇒ déploiement d’une méthode “resetCharge”, permettant de réinitialiser les zones.

⇒ déploiement d’une méthode “add” permettant d’ajouter un aliment récupéré en base de données, avec sa charge glycémique, et le grammage.

⇒ mise en place d’un message d’erreur si les zones ne sont pas renseignées.

⇒ déploiement d’une méthode permettant de créer un menu en le nommant + message d’erreur en cas de non saisie.

⇒ mise en place de boutons (“click”) permettant d’ajouter (par la méthode “add”) les aliments.

*Pour la partie HTML:*

⇒ mise en place d’un “mat-autocomplete” pour la recherche d’un aliment

⇒ d’une zone pour la charge glycémique, du poids.

***7/ Le component history-meals:***

Il correspond à la page “Meals history” de l’application Nutrapp.

⇒ Pour la partie HTML: mise en place d’un tableau avec plusieurs colonnes renseignées en dur avec les aliments, la date, le repas de la journée, la composition du menu et la charge glycémique.

8/ ***Le component list-aliment:***

Il correspond à la page Aliment list de l’application Nutrapp et permet de gérer l’affichage des aliments récupérés dans la base de données et l’affichage de leur charge glycémique.

*Développement de la partie BACK de l’application NUTRAPP*

Le framework utilisé pour la partie développement JAVA est **SPRING**, nous avons choisi d’utiliser **JPA/HIBERNATE** pour le management de nos objets.

Le lien git du projet est le suivant :<https://github.com/Waby1/NutrappV1>

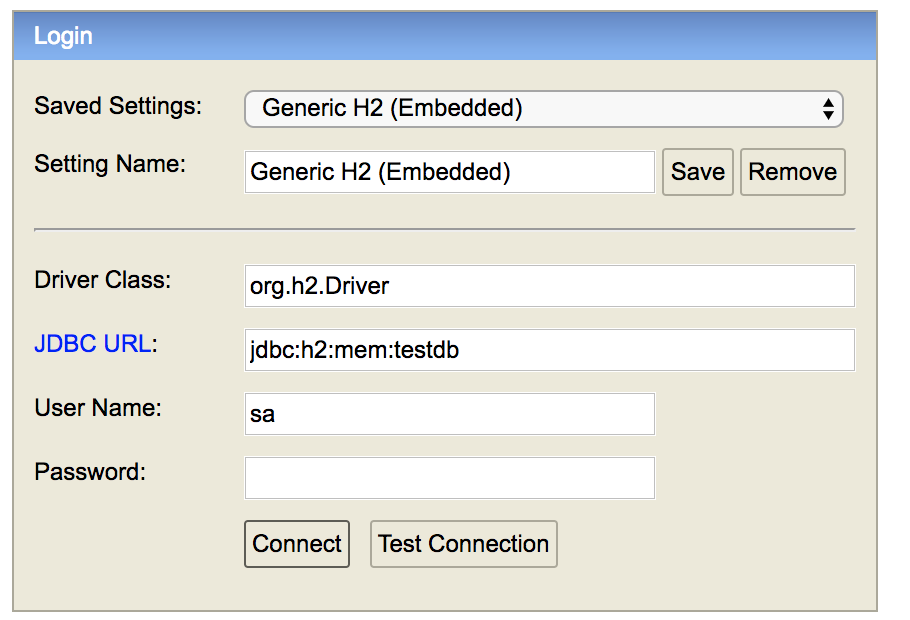
Le type de base de données choisi est une base H2

Le port d’écoute de celle ci est le 8090,

Pour accéder à la partie administration / consultation il faudra utiliser l’url

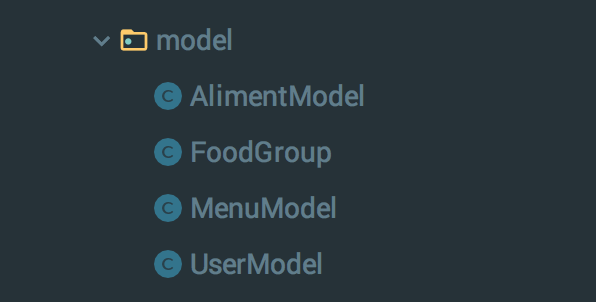
<http://localhost:8090/h2-console/>

avec les id ci dessous :



**Les Model**

Nous avons créé différents models des objets que nous allons utiliser.



L’AlimentModel qui définit la composition de nos objets aliments formés d’un ID généré de manière automatique en base, d’un nom, d’une charge glycémique et d’un FoodGroup.

**public class AlimentModel {**

**@Id**

**@GeneratedValue(**strategy **=** GenerationType**.*AUTO*)**

**private Long** id**;**

**@Column (**name **=** "name"**)**

**private String** name**;**

**@Column (**name **=** "chargeGly"**)**

**private long** chargeGly**;**

**@ManyToOne(**fetch **=** FetchType**.*LAZY*)**

**@JoinColumn(**name **=** "FK\_FOODGROUP"**)**

**private FoodGroup** foodGroup**;**

Le MenuModel qui définit notre objet menu formé d’un ID généré de manière automatique en base, d’un nom, d’une charge glycémique et d’une liste d’aliment en jointable

**public class MenuModel {**

**@Id**

**@GeneratedValue(**strategy **=** GenerationType**.*AUTO*)**

**private Long** id**;**

**@Column(**name **=** "name"**)**

**private String** name**;**

**@Column(**name **=** "ChargeGlyMeals"**)**

**private long** chargeGlyMeals**;**

**@ManyToMany(**cascade **=** CascadeType**.*ALL*)**

**@JoinTable(**name **=** "menu\_composition"**,** joinColumns **= {@JoinColumn(**name **=** "FK\_ALIMENT"**,** referencedColumnName **=** "id"**)},**

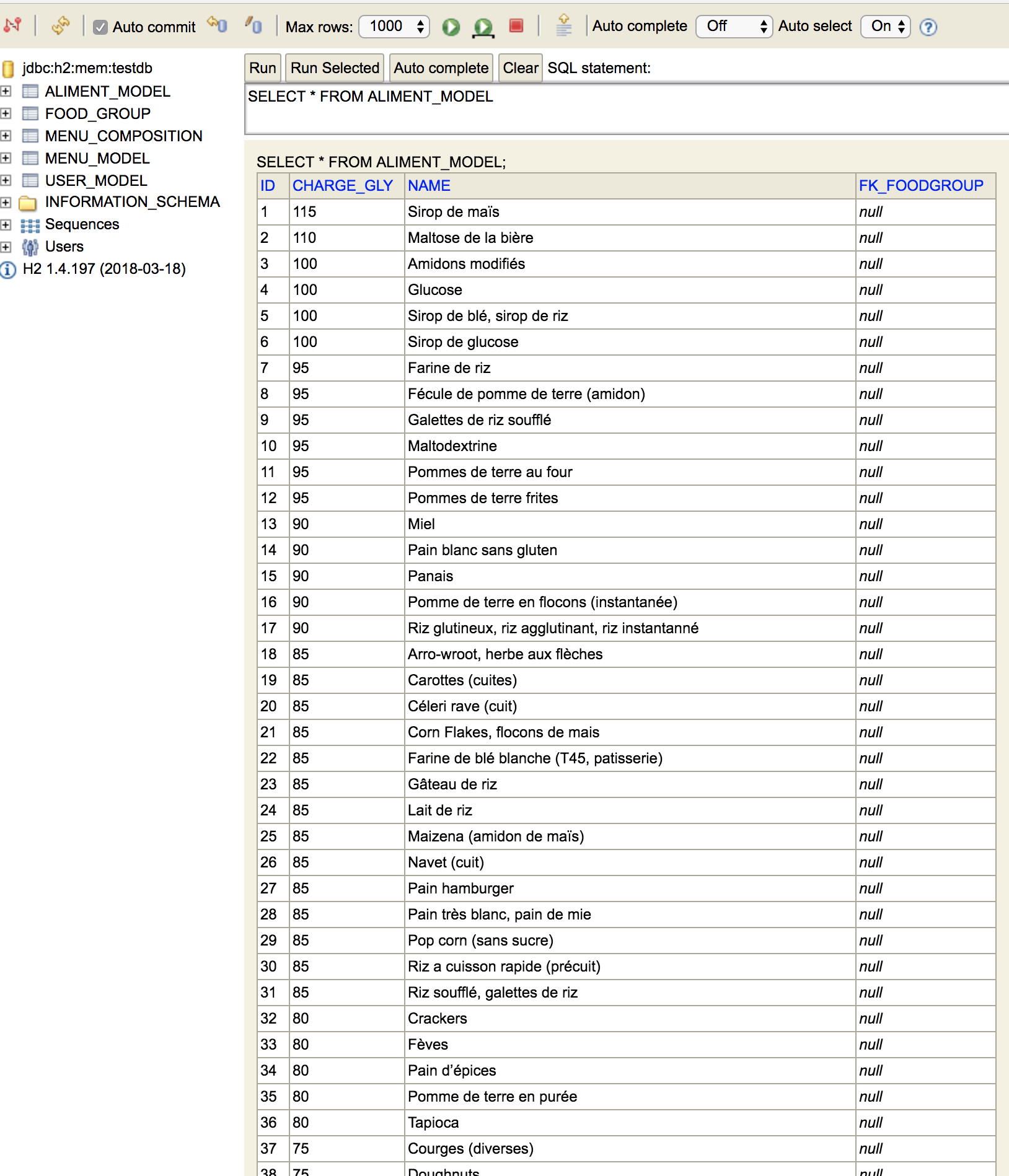
inverseJoinColumns **= {@JoinColumn(**name **=** "FK\_MENU"**,** referencedColumnName **=** "id"**)}**

**)**

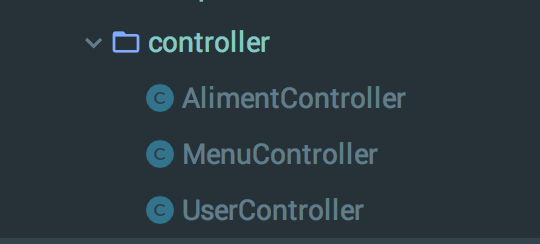
**private** List**<AlimentModel>** menuComposition **= new** ArrayList**<AlimentModel>();**

Le UserModel et le FoodGroup n’ont pas été utilisé dans la version finale de l’application, puisque nous ne gérons pas la connexion utilisateur, ni le classement affichage des aliments par type.

Dans le dossier H2Folders une fonction a été implémentée pour charger au lancement de l’application notre base de données alimentaire à partir d’un fichier d’un fichier Excel,



**Les Controller**



Nous avons 3 controllers, le UserController n’est pas utilisé dans notre application,

AlimentController est utilisé pour restituer l’ensemble des aliments présents en BDD qui seront affichés dans la partie front appelée par le http://localhost:4200.

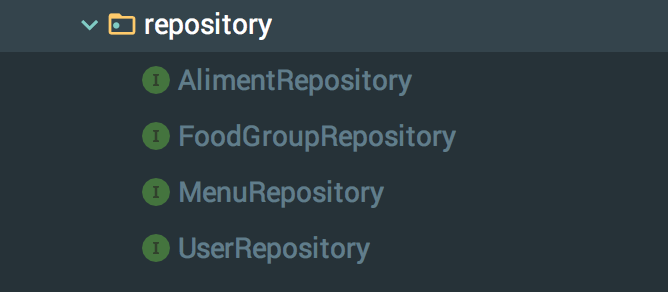
**@Controller**

**@RequestMapping(**"/api/aliments"**)**

**@CrossOrigin(**origins **=** "http://localhost:4200"**)**

Le MenuController va permettre d’enregistrer le menu généré par notre application en front, nous passerons pour cela par une classe MenuService qui permettra de persister les objets, celle ci est définie avec une annotation @transactional qui permettra un rollback immédiat en cas de plantage.

**Les Repository**



Les repository vont stocker nos menus et nos aliments, le FoodGroupRepository et le UserRepository ne seront pas utilisés dans notre application.